Introduction to Artificial Intelligence (AI)

Computer Science cpsc502, Lecture 11

Oct, 18, 2011

Planning in Stochastic Environments

Environment

Deterministic | Stochastic

Problem

Static
- Constraint Satisfaction
- Query

Sequential
- Planning

Problem	Deterministic	Stochastic
Constraint Satisfaction	*Vars + Constraints* — Arc Consistency, Search, SLS	
Query	*Logics* — Search	*Belief Nets* — Var. Elimination; *Markov Chains and HMMs*
Planning	*STRIPS* — Search	*Decision Nets* — Var. Elimination; *Markov Processes* — Value Iteration

for CSP

CSP

for Inference

CSP

for complex planning

Representation

Reasoning Technique

Planning Under Uncertainty: Intro

- **Planning** how to select and organize a sequence of actions/decisions to achieve a given goal.
- **Deterministic Goal**: A possible world in which some propositions are assigned to T/F

- **Planning under Uncertainty**: how to select and organize a sequence of actions/decisions to "*maximize the probability*" of *"achieving a given goal"*
- **Goal under Uncertainty**: we'll move from all-or-nothing goals to a richer notion: rating how *happy* the agent is in different possible worlds.

“Single” Action vs. Sequence of Actions

Set of primitive decisions that can be treated as a single macro decision to be made *before acting*

one-off

- Agents makes observations
- Decides on an action
- Carries out the action

Sequential Decisions

Today Oct 18

One-Off Decisions

- **Utilities / Preferences and optimal Decision**
- **Single stage Decision Networks**

Sequential Decisions

- Representation
- Policies
- Finding Optimal Policies

One-off decision (textbook example)

Delivery Robot Example

not

A true
false

- Robot needs to reach a certain room
- Going through stairs may cause an accident.
- It can go the short way through long stairs, or the long way through short stairs (that reduces the chance of an accident but takes more time)

Which way long
short

$$P(A=t \mid WW=long) < P(A=t \mid WW=short)$$

- The Robot can choose to wear pads to protect itself or not (to protect itself in case of an accident) but pads slow it down

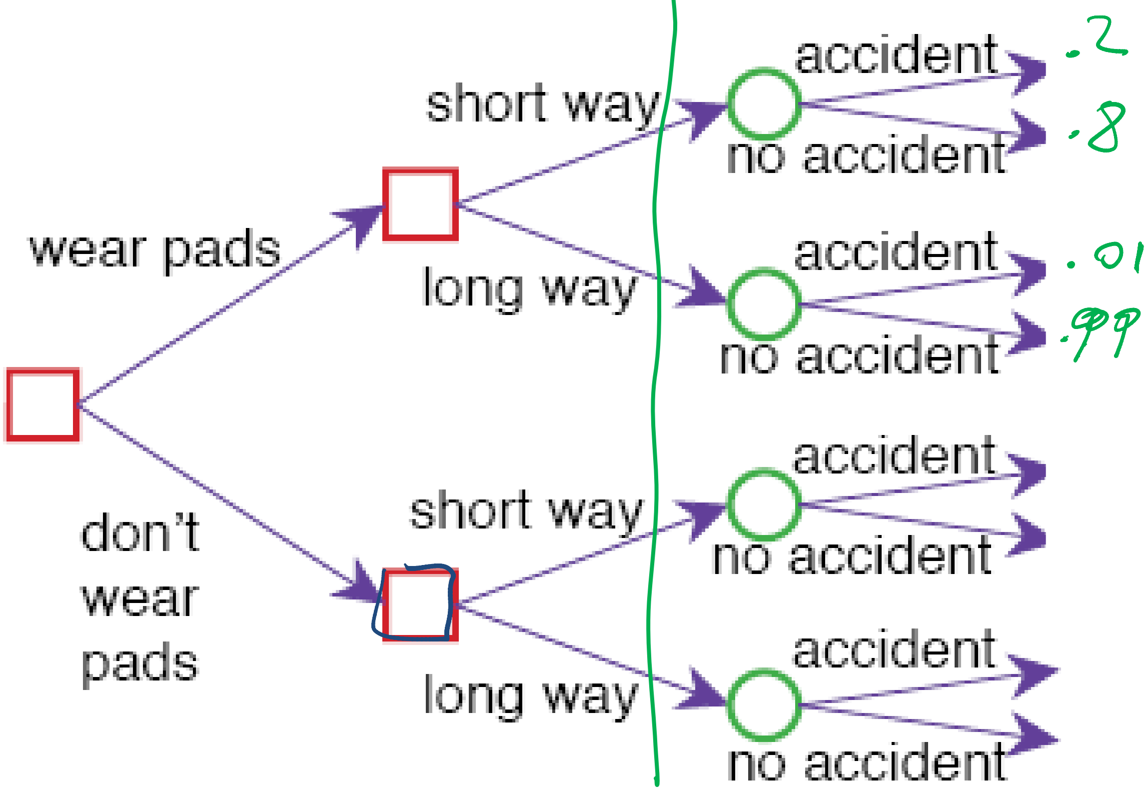
Wear pads t
f

if $A=t$

- If there is an accident the Robot does not get to the room

Decision Tree for Delivery Robot

- This scenario can be represented as the following decision tree

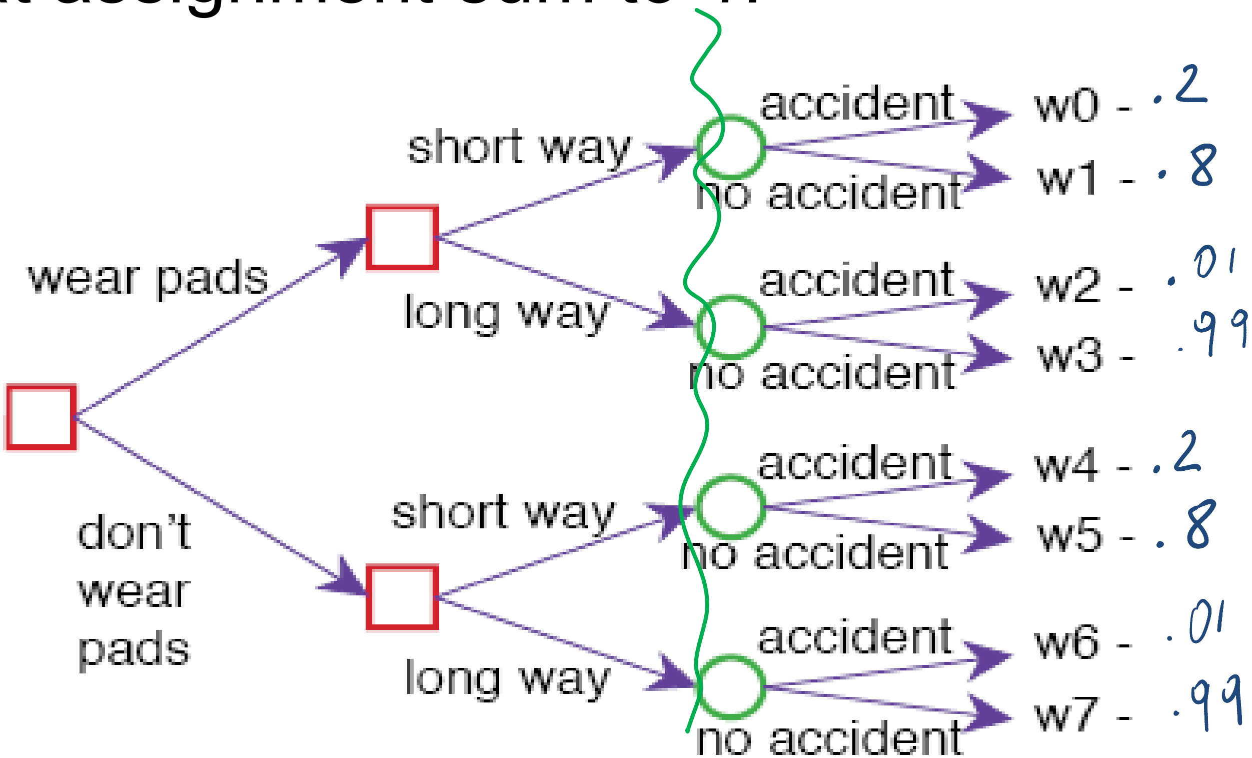


Which way	Accident	
long	true	0.01
long	false	0.99
short	true	0.2
short	false	0.8

- The agent has a set of decisions to make (a macro-action it can perform)
- Decisions can influence random variables
- Decisions have probability distributions over outcomes

Decision Variables: Some general Considerations

- A possible world specifies a value for each random variable and each decision variable.
- For each assignment of values to all decision variables, the probabilities of the worlds satisfying that assignment sum to 1.

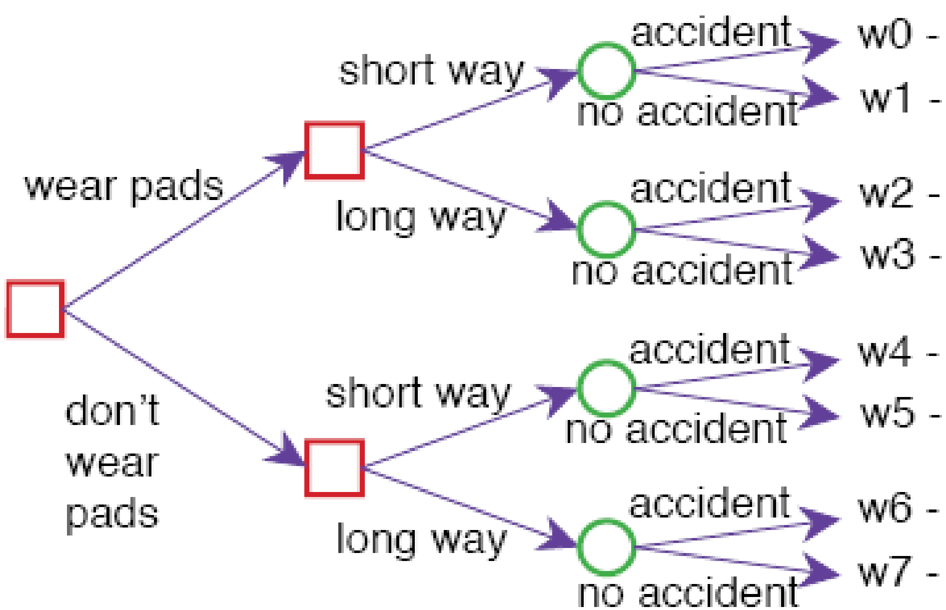


Slide 8

What are the optimal decisions for our Robot?

It all depends on how happy the agent is in different situations.

For sure getting to the room is better than not getting there….. but we need to consider other factors..



Utility / Preferences

Utility: a measure of desirability of possible worlds to an agent

- Let *U* be a real-valued function such that *U* (*w*) represents an agent's degree of preference for world *w* . [0, 100]

Would this be a reasonable utility function for our Robot?

Which way	Accident	Wear Pads	Utility	World
short	true	true	35	w0, moderate damage
short	false	true	95	w1, reaches room, quick, extra weight
long	true	true	30	w2, moderate damage, low energy
long	false	true	75	w3, reaches room, slow, extra weight
short	true	false	3	w4, severe damage
short	false	false	100	w5, reaches room, quick Best
long	true	false	0	w6, severe damage, low energy
long	false	false	80	w7, reaches room, slow

Utility: Simple Goals

- Can simple (boolean) goals still be specified?

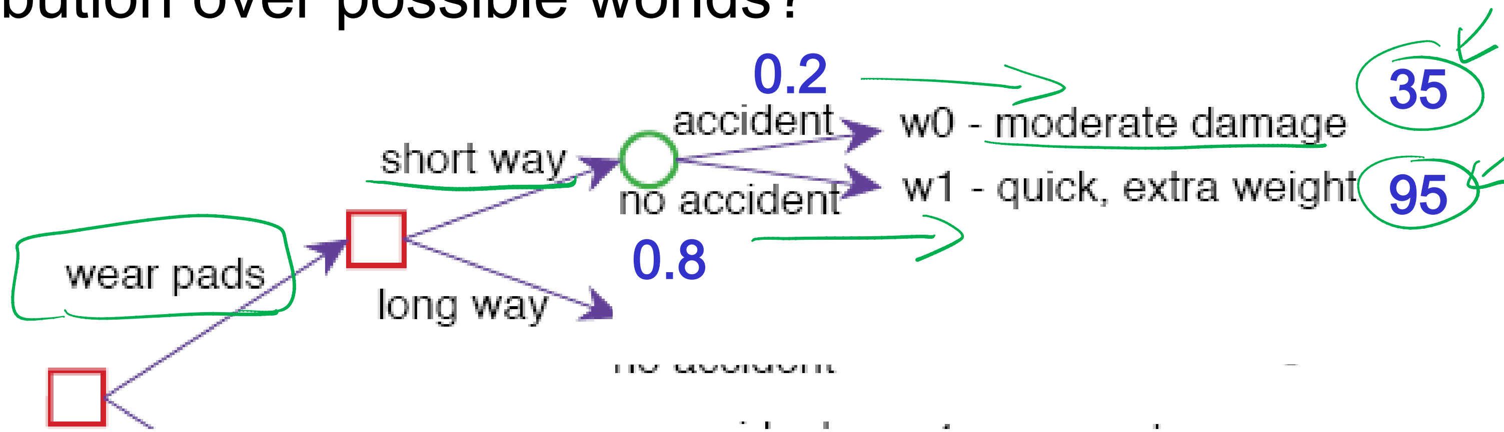
goal: "reaching the room"

Accident must be false

Which way	Accident	Wear Pads	Utility
long	true	true	0
long	true	false	0
long	false	true	100
long	false	false	100
short	true	true	0
short	true	false	0
short	false	true	100
short	false	false	100

Optimal decisions: How to combine Utility with Probability

What is the utility of achieving a certain probability distribution over possible worlds?



- It is its expected utility/value i.e., its average utility, weighting possible worlds by their probability.

EU(wp=t, ww=short) =

.2 * 35 + .8 * 95

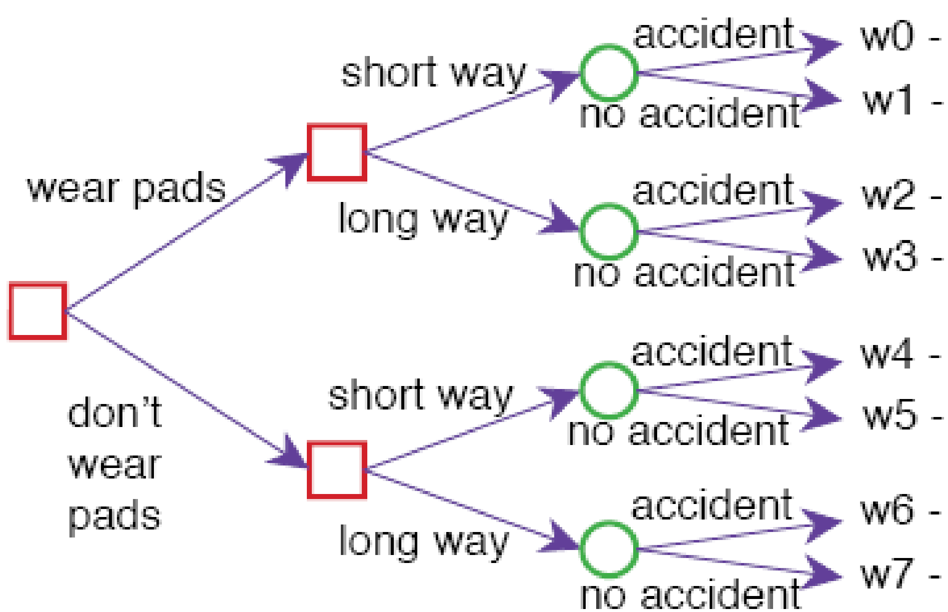
Optimal decision in one-off decisions

- Given a set of n decision variables var_i (e.g., Wear Pads, Which Way), the agent can choose:

  $D = d_i$; d_i in dom(var_1) x .. x dom(var_n) .

d_i

Wear Pads	Which way
true	short
true	long
false	short
false	long

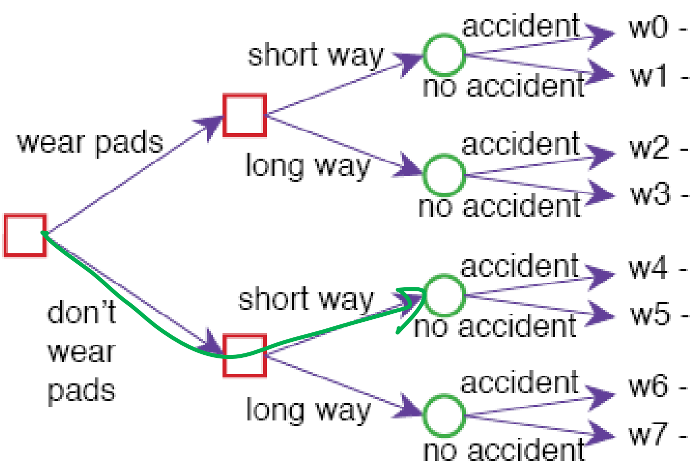


Optimal decision: Maximize Expected Utility

- The expected utility of decision $D = d_i$ is

$$\mathbb{E}(U \mid D = d_i) = \sum_{w \models D = d_i} \boxed{P(w \mid D = d_i)}\ \boxed{U(w)}$$

e.g., $\mathbb{E}(U \mid D = \{WP=\text{false}\ ,\ WW=\text{short}\ \}) =$



$P(w_4) * U(w_4) +$
$P(w_5) * U(w_5)$

- An optimal decision is the decision $D = d_{max}$ whose expected utility is maximal:

$$d_{max} = \arg\max_{d_i \in dom(D)} \mathbb{E}(U \mid D = d_i)$$

max

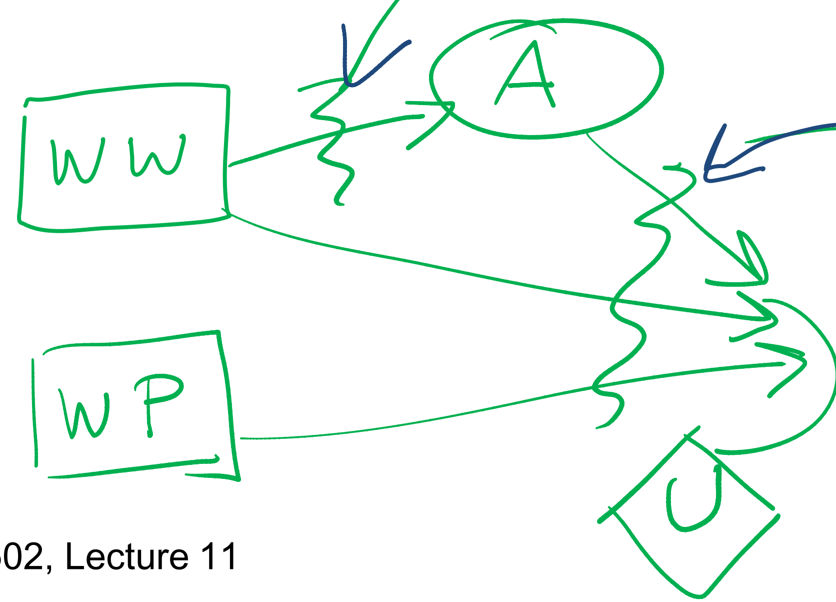
Wear Pads	Which way	EU
true	short	–
true	long	–
false	short	–
false	long	–

Single-stage decision networks

Extend belief networks with:

- **Decision nodes**, that the agent chooses the value for. Drawn as rectangle.
- **Utility node**, the parents are the variables on which the utility depends. Drawn as a diamond.
- Shows explicitly which decision nodes affect random variables

Which way	Accident	
long	true	0.01
long	false	0.99
short	true	0.2
short	false	0.8



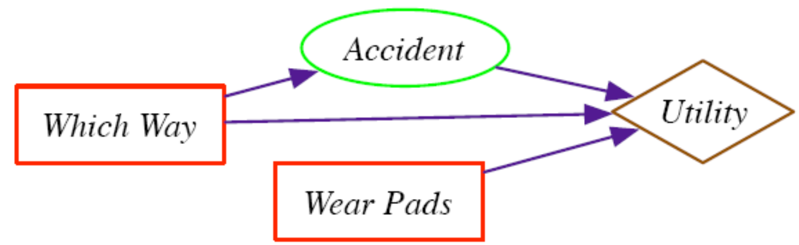
Which way	Accident	Wear Pads	Utility
long	true	true	30
long	true	false	0
long	false	true	75
long	false	false	80
short	true	true	35
short	true	false	3
short	false	true	95
short	false	false	100

Finding the optimal decision: We can use VE

Suppose the random variables are $X_1, \ldots, X_n$, the decision variables are the set D, and utility depends on

$pU \subseteq \{X_1, \ldots, X_n\} \cup D$

Parents of U



$$\mathbb{E}(U|D) = \sum_{X_1,\ldots,X_n} P(X_1,\ldots,X_n|D)\,U(pU)$$

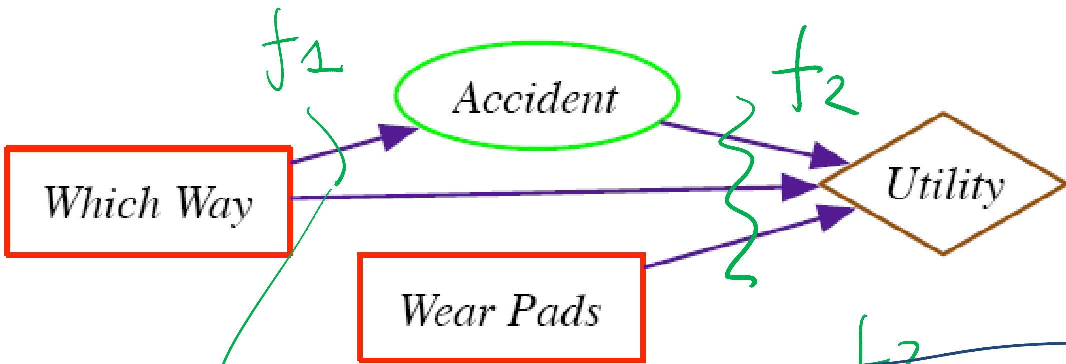
$$= \sum \prod P(X_i|pX_i)\;U(pU)$$

also includes decision vars

To find the optimal decision we can use VE:

1. Create a factor for each conditional probability and for the utility
2. Multiply factors and sum out all of the random variables (This creates a factor on D that gives the expected utility for each d_i)
3. Choose the d_i with the maximum value in the factor.

Example Initial Factors (Step1)



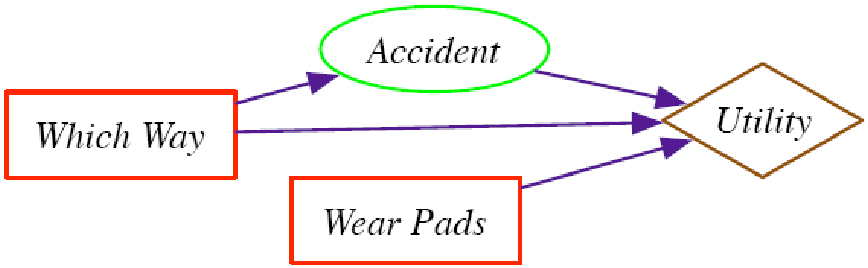
f1

Which way	Accident	Probability
long	true	0.01
long	false	0.99
short	true	0.2
short	false	0.8

f2

Which way	Accident	Wear Pads	Utility
long	true	true	30
long	true	false	0
long	false	true	75
long	false	false	80
short	true	true	35
short	true	false	3
short	false	true	95
short	false	false	100

Example: Multiply Factors (Step 2a)



$$\sum_A \overset{f}{\boxed{f_1(WW, A) \times f_2(A, WW, WP)}}$$

f_1

Which way	Accident	Probability
long	true	0.01
long	false	0.99
short	true	0.2
short	false	0.8

f_2

Which way	Accident	Wear Pads	Utility
long	true	true	30
long	true	false	0
long	false	true	75
long	false	false	80
short	true	true	35
short	true	false	3
short	false	true	95
short	false	false	100

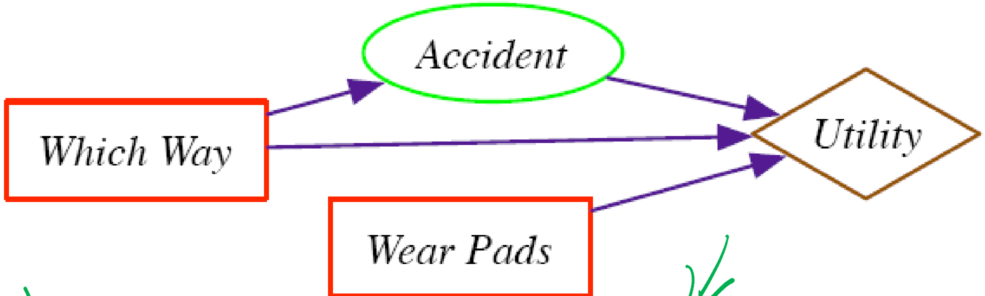
f_3

Which way	Accident	Wear Pads	Utility
long	true	true	30 * .01
long	true	false	0 × .01
long	false	true	75 × .99
long	false	false	80
short	true	true	35
short	true	false	3
short	false	true	95
short	false	false	100

WW WP

t t

Example: Sum out vars and choose max (Steps 2b-3)



$$\sum_{A} f'(A, WW, WP)$$

Sum out accident:

Which way	Accident	Wear Pads	Utility
long	true	true	0.01*30
long	true	false	0.01*0
long	false	true	0.99*75
long	false	false	0.99*80
short	true	true	0.2*35
short	true	false	0.2*3
short	false	true	0.8*95
short	false	false	0.8*100

Which way	Wear Pads	Expected Utility
long	true	0.01*30+0.99*75=74.55
long	false	0.01*0+0.99*80=79.2
short	true	0.2*35+0.8*95=83
short	false	0.2*3+0.8*100=80.6

Thus the optimal policy is to take the short way and wear pads, with an expected utility of 83.

Today Oct 18

<u>One-Off Decision</u>

- Utilities / Preferences and optimal Decision
- Single stage Decision Networks

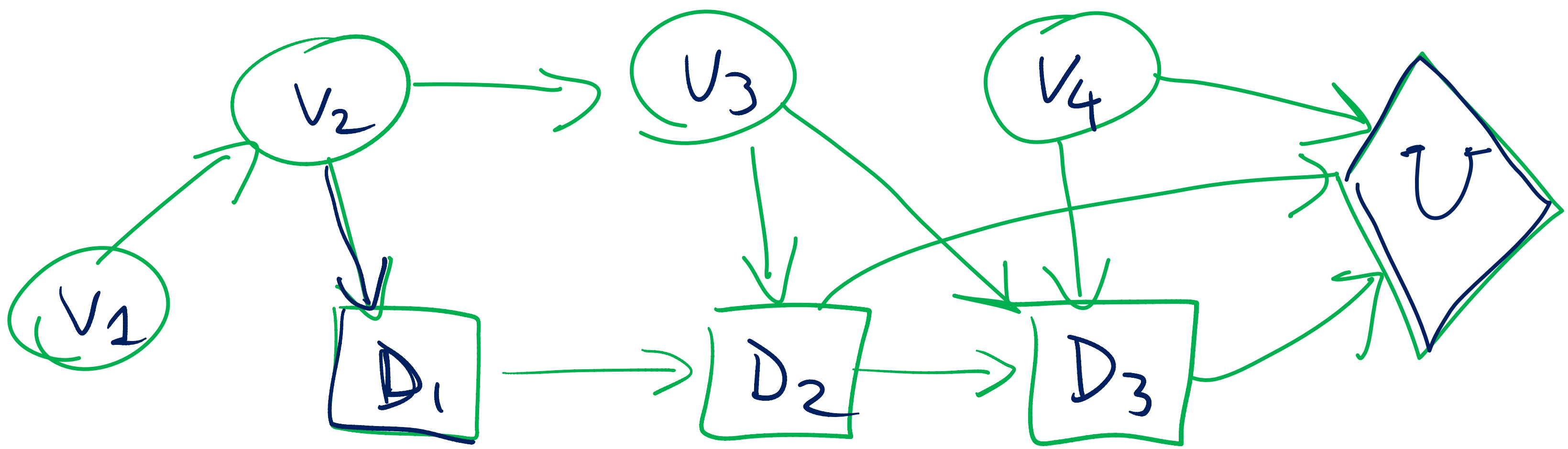
<u>**Sequential Decisions**</u>

- **Representation**
- **Policies**
- **Finding Optimal Policies**

Sequential decision problems

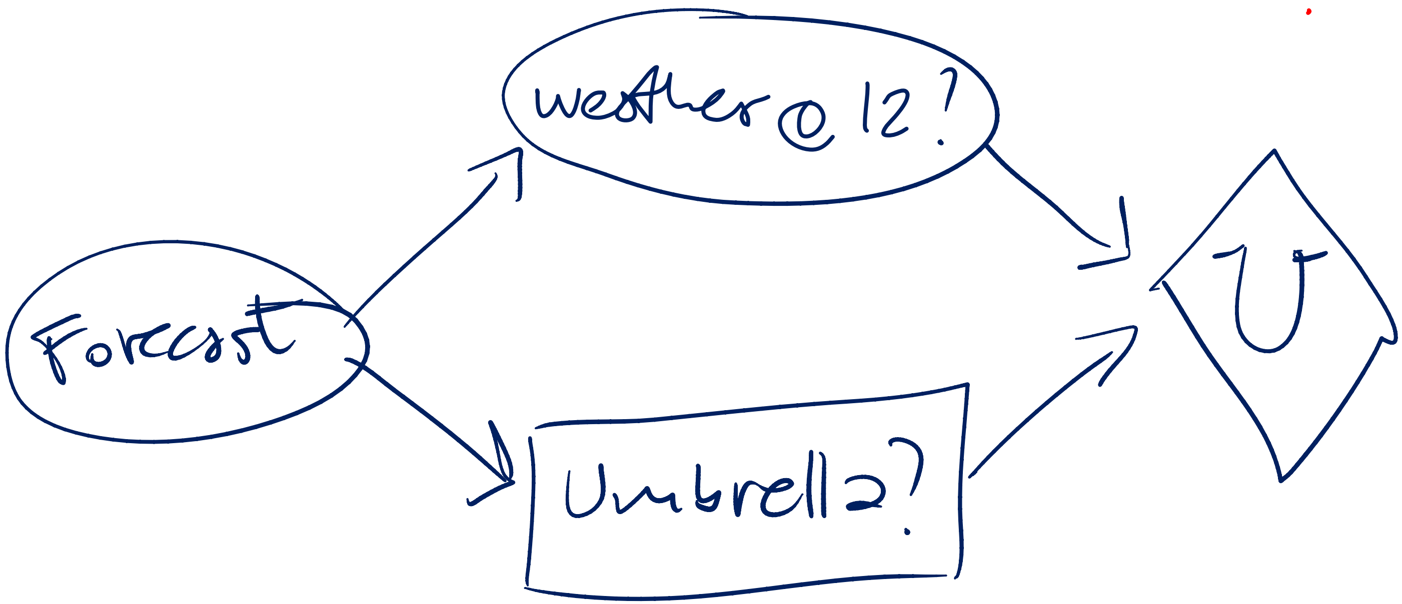
- A sequential decision problem consists of a sequence of decision variables $D_1, \ldots\ldots, D_n$.
- Each D_i has an information set of variables pD_i, whose value will be known at the time decision D_i is made.

$pD_3 = \{D_2\ V_3\ V_4\}$



Sequential decisions : Simplest possible

- Only one decision! (but different from one-off decisions)
- Early in the morning. Shall I take my **umbrella** today? (I'll have to go for a long walk at noon)
- Relevant Random Variables?



Policies for Sequential Decision Problem: Intro

- A **policy** specifies what an agent should do under each circumstance (for each decision, consider the parents of the decision node)

In the *Umbrella* "degenerate" case:

D_1 ? T F

Some possible Policy

pD_1				
Rainy	→ R	T	F	T ...
Cloudy	→ C	T	F	T ...
Sunny	→ S	F	F	T ...

3 policies

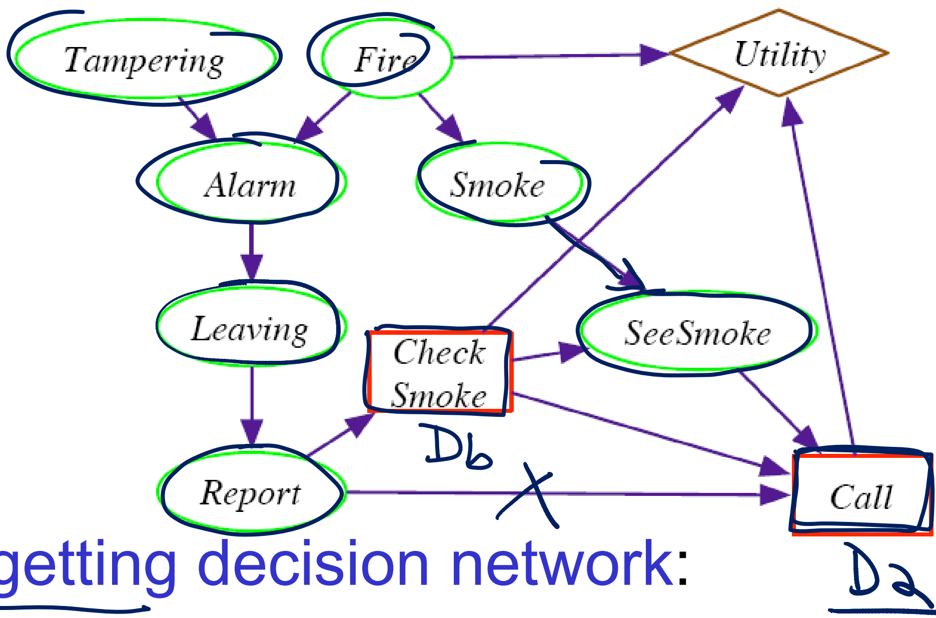
How many policies?

$\boxed{2}$ $\boxed{3}$

$|D|^{|pD|}$

Sequential decision problems: "complete" Example

- A sequential decision problem consists of a sequence of decision variables $D_1, \ldots\ldots, D_n$.
- Each D_i has an information set of variables pD_i, whose value will be known at the time decision D_i is made.



$pCS = \{R\}$

$pC = \{R, CS, SS\}$

No-forgetting decision network:

- decisions are totally ordered
- if a decision D_b comes before D_a, then
  - D_b is a parent of D_a
  - any parent of D_b is a parent of D_a

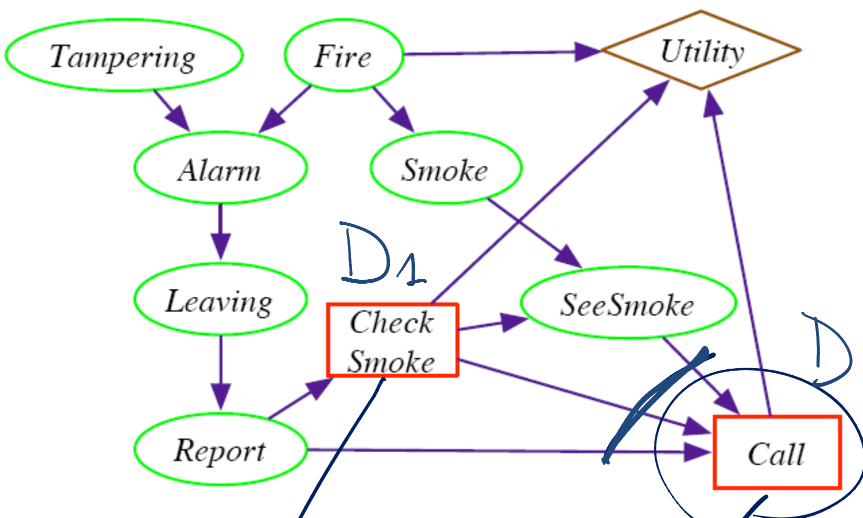
$pCS \subseteq pC$

Policies for Sequential Decision Problems

- A **policy** is a sequence of $\delta_1, \ldots\ldots, \delta_n$ **decision functions**

$$\delta_i : \text{dom}(pD_i) \rightarrow \text{dom}(D_i)$$

- This policy means that when the agent has observed $O \in \text{dom}(pD_i)$, it will do $\delta_i(O)$



Example: δ_1

Report	Check Smoke
T	T ~~F F F~~
F	T ~~F T F~~

δ_2

Report	CheckSmoke	SeeSmoke	Call
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

8

How many policies?

$2^2 * 2^8$

When does a possible world satisfy a policy?

- A possible world specifies a value for each random variable and each decision variable.
- **Possible world w satisfies policy δ**, written $w \models \delta$ if the value of each decision variable is the value selected by its decision function in the policy (when applied in w).

δ

$w \not\models \delta$

w

Rnd.

Dec.

VARs	
Fire	true
Tampering	false
Alarm	true
Leaving	true
Report	false
Smoke	true
SeeSmoke	true
CheckSmoke	true
Call	true

Decision function for…

Report	**Check Smoke**
true	true
false	false

$\not\models$

Decision function for…

Report	**CheckSmoke**	**SeeSmoke**	**Call**
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

When does a possible world satisfy a policy?

- Possible world *w* satisfies policy *δ*, written $w \models \delta$ if the value of each decision variable is the value selected by its decision function in the policy (when applied in *w*).

$w_1 \models \delta$

VARs	
Fire	true
Tampering	false
Alarm	true
Leaving	true
Report	true
Smoke	true
SeeSmoke	true
CheckSmoke	true
Call	true

δ

Decision function for…

Report	**Check Smoke**
true	true
false	false

✓

Decision function for…

Report	**CheckSmoke**	**SeeSmoke**	**Call**
true	true	true	true
true	true	false	false
true	false	true	true
true	false	false	false
false	true	true	true
false	true	false	false
false	false	true	false
false	false	false	false

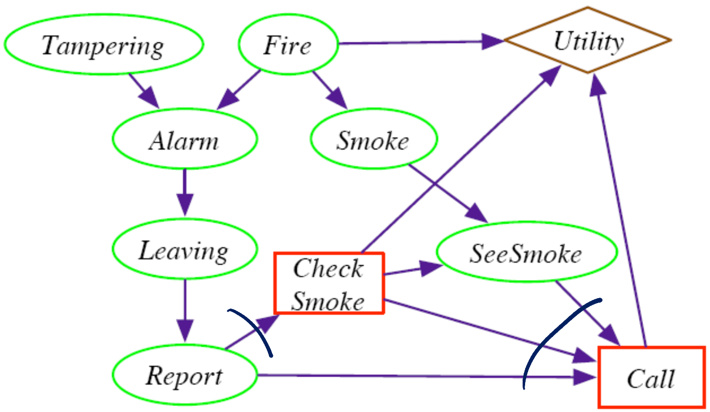
Expected Value of a Policy

- Each possible world w has a probability $P(w)$ and a utility $U(w)$
- The expected utility of policy δ is

$$\sum_{w \models \delta} P(w) * U(w)$$

- The optimal policy is one with the max expected utility.

Complexity of finding the optimal policy: how many policies?



- How many assignments to parents?

  CS 2 C 2^3

- How many decision functions? (binary decisions)

  2^{2} 2^{2^3}

- How many policies? product $2^{2} * 2^{2^3}$

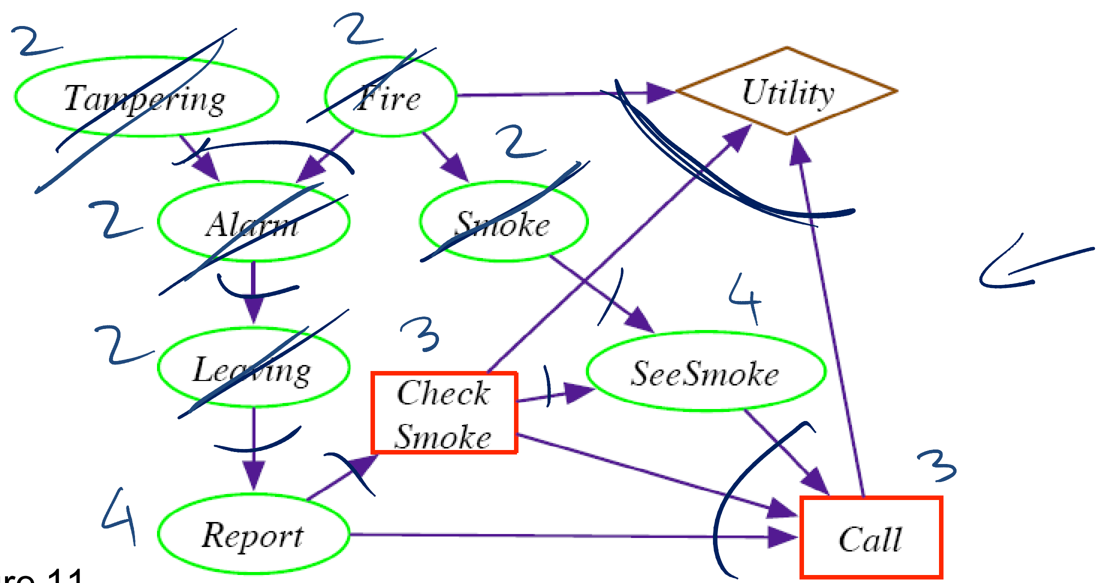
- If a decision *D* has *k* binary parents, how many assignments of values to the parents are there? 2^K

- If there are *b* possible actions (possible values for D), how many different decision functions are there? $\boxed{b^{2^K}}$

- If there are *d* decisions, each with *k* binary parents and *b* possible actions, how many policies are there? $\left(b^{2^K}\right)^d$

Finding the optimal policy more efficiently: VE

1. Create a factor for each conditional probability table and a factor for the utility.
2. **Sum out** random variables that are not parents of a decision node.
3. **Eliminate** (aka sum out) the decision variables
4. **Sum out** the remaining random variables.
5. **Multiply the factors**: this is the expected utility of the optimal policy.



Eliminate the decision Variables: step3 details

- Select a variable D that corresponds to the latest decision to be made
  - this variable will appear in only one factor with (some of) its parents
- Eliminate D by maximizing. This returns:
  - The optimal decision function for D, $\arg\max_D f$
  - A new factor to use in VE, $\max_D f$
- Repeat till there are no more decision nodes.

Example: Eliminate CheckSmoke

Report	CheckSmoke	Value
true	true	-5.0
true	false	-5.6
false	true	-23.7
false	false	-17.5

New factor

Report	Value
true	-5.0
false	-17.5

Decision Function

Report	CheckSmoke
true	true
false	false

VE elimination reduces complexity of finding the optimal policy

- We have seen that, if a decision *D* has *k* binary parents, there are *b* possible actions, If there are d decisions,
- Then there are: $(b^{2^k})^d$ *policies*
- Doing variable elimination lets us find the optimal policy after considering only $d \, . b^{2^k}$ policies (we eliminate one decision at a time)
  - VE **is much more efficient** than searching through policy space.
  - However, this complexity is **still doubly-exponential** we'll only be able to handle relatively small problems.

+ give up nonforgetting assump
+ approx. algorithms

Return Assignment-1

Tot. Count 14 – max 94%; min 43%; avg 72%

6 below 70% 3 below 50%

TODO for this Thurs

- **Finish Assignment2 (last question)**

- Also Do exercises 9.A and 9.B

http://www.aispace.org/exercises.shtml

These two exercises are going to help you a lot with the assignment question ;-)